递归：递归属于CPU密集型的算法，对CPU要求比较高，JS属于单线程，只使用一个CPU内核。JS不适合递归中深层次嵌套。

使用递归：删除某个目录下的所有文件（不确定有多少文件）

练习：创建一个函数，传递任意一个数字，使用递归计算1-该数字之间所有整数的和。n-->1

匿名函数：定义时未直接指定名称的函数。

function (){

}

①创建函数

//函数声明

function fn(){ }

console.log(fn) ===>function fn

//函数表达式

var fun = function(){ }

console.log(fun) ====>function fun

函数表达式所创建的函数时通过变量创建，只存在变量的声明和提升，不存在函数提升，所以必须先创建在调用。

对比函数名称和函数名称()

函数名称() 调用函数，执行函数体中的代码，得到函数的返回结果。

函数名称 本质是一个变量，保存了一个函数。

全局污染：全局变量的出现影响到了其他的变量，（全局变量：var a = 1;创建的函数也是全局变量），避免全局污染：使用局部变量

②匿名函数的自调用（防止全局污染）

(function(){

var a = 1; //创建的作用域，防止污染全局

})();

//把全局变量a转为局部

(function () {

var num = 1;

console.log(num) //1

})();

(function () {

var num = 2;

console.log(num) //2

})();

③回调函数：将函数以实参的形式（函数名称或者匿名函数）传递叫做回调函数，通过参数来调用。

// 回调函数

function ran(madai) {//形参，用于接收所传递的函数

console.log("开始====");

console.log("结束=====");

// 调用传递的回调函数

//madai()===>dong()

madai();

}

function dong() {

console.log("kaishi..");

console.log("jieshu..")

}

function \_dong() {

console.log("1111111")

}

//dong以实参形式传递进去，叫回调函数

// ran(dong)

ran(\_dong)

// 匿名函数以实参形式传递，叫回调函数

// ran(function(){return 1;})

function getRes(fun, fun2) {

console.log(fun(100))

console.log(fun2())

}

//1-00和 回调函数

function getSum(n) {

for (var i = 1, sum = 0; i <= n; i++) {

sum += i;

}

return sum;

}

//1-10阶乘 回调函数

function getChen() {

for (var i = 1, res = 1; i <= 10; i++) {

res \*= i;

}

return res;

}

//回调函数 getSum,getChen

getRes(getSum, getChen)

// getRes(getChen)

④系统函数 (全局函数Number、parseInt、parseFloat)：

isNaN：用于检测一个值转为数值后是否为NaN，会自动调用Number

var str = '12a';

console.log( Number(str) );

检测是否为NaN

console.log(isNaN(str))

是NaN===>true

不是NaN====>false

isFinite：检测一个值是否是有限值。只有无穷Infinity是无限值。是有限值返回true，不是有限值返回false

0做除数返回Infinity

eval() 执行字符串中的表达式。

对象：

对象属于引用类型数据

对象是一组属性和方法的集合

万物皆对象

1、JS中的对象。

自定义对象：用户创建的对象。

宿主对象：根据不同的执行环境划分的，

内置对象：JS提供的对象

自定义对象：

创建：对象字面量、内置构造函数创建、自定义构造函数

使用对象字面量创建对象。

使用大括号 { } 创建对象，属性由属性名和属性值,之间用冒号隔开，多组属性之间用逗号隔开，属性名的引号可加可不加，如果含有特殊字符必须加引号。

// 创建员工对象编号，姓名，性别，生日，工资，所属部门

var emp = {

eid: 1001,

ename: '张三',

sex: '男',

birthday: '1995-06-07',

salary: 9999.00,

dept\_name: '研发部',

'dept-id':20,

};

// 重新赋值

emp.sex = '女'

// 访问对象的属性

console.log(emp.ename)

// 访问对象中属性含有特殊字符的属性

console.log(emp['dept-id'],emp['ename'])

访问对象的属性：对象名.属性名,含有特殊字符时dept-id==>emp['dept-id'],属性不存在，返回undefined

访问对象的方法：对象名.方法名();

使用内置构造函数创建对象。

var student = new Object();

创建一个空对象，需要单独添加每一个属性。

访问对象的所有属性。

for(var key in car){//key代表对象的每一个属性名。对象[key]代表每一个属性名对应的属性值

key==>对象中每一个属性名;

car==>遍历的对象。

console.log(car[key]); //==>如果使用car.key访问对象属性，会把key当做对象的属性，结果undefined

}